Attorney Docket No.: 16869P-085500
Client Ref. No. 340200946US1

PATENT APPLICATION

Storage Device and Method of Setting Configuration Information of Same

Inventors: **Toshimichi KISHIMOTO**
Residence: Hadano, Japan
Citizenship: Japan

Yoshinori IGARASHI
Residence: Odawara, Japan
Citizenship: Japan

Shuichi YAGI
Residence: Matsuda, Japan
Citizenship: Japan

Assignee: **Hitachi, Ltd.**
6, Kanda Surugadai 4-chome
Chiyoda-ku, Tokyo, Japan
Incorporation: Japan

Entity: Large

TOWNSEND and TOWNSEND and CREW LLP
Two Embarcadero Center, 8th Floor
San Francisco, California 94111-3834
Tel: 650-326-2400

Storage Device and Method of Setting Configuration Information of Same

CROSS-REFERENCE TO RELATED APPLICATION

[0001] This application claims the right of priority based on Japanese Patent Application No. 2002-303718 filed on October 18, 2002, and cites the application in the specification of this application.

BACKGROUND OF THE INVENTION

Field of the Invention

[0002] The present invention relates to a method of setting the configuration information of an information processing apparatus, and more particularly, to setting and modifying storage configuration information. The invention also relates to a storage device capable of setting such configuration information.

Description of the Related Art

[0003] Information processing devices and the like in system environments continually become more complex, and various proposals have been presented to simplify the setting operations of such complex information processing devices. Setting the configuration information (for example, path definition, creation of logic volume, physical format, security information) of a storage device in an information processing system involves the risk of data loss due to an increase in the number of setting operations and in wrong settings made by supervisors, because the devices have become large and complex.

[0004] JP-A-5-128032 proposes, as a method of solving such problems, a technique for settings in a network environment, which can be set by end users of average skill. Another Japanese patent application, JP-A-6-175827, proposes a technique of automatically creating settings of a network environment in which a program is implemented on a computer, to eliminate the need for end users to make the difficult settings for an environment. Further, JP-A-11-161604 discloses a technique in which the settings for a network environment with connected clients are preserved as a script file in a server connected to the network and automatically downloaded for setting the environment.

[0005] Even with a Graphical User Interface (GUI), setting modifications can cause unexpected equipment trouble even with supervisors performing the setting operations. To

prevent such unauthorized operation, JP-A-2000-181687 discloses a method in which system management information is preserved in two memories and errors are avoided by referring to one of the memories when an error occurs after system management information in the other memory is edited. JP-A-2000-181687 also discloses a method in which a password is required when setting information is edited. These methods ensure that system management information is safely preserved.

[0006] However, when the settings for the environment and structure of an information processing device are modified, operations in the modified environment are sometimes incompatible with those performed in the previous environment and structure. For example, when the structure of storage areas in a storage device is modified, concern arises that the operations may call for data being written into areas in positions essentially unsuitable for writing.

[0007] In particular, in structural modification, such as setting the storage areas of a storage device, the integrity of written data must also be guaranteed for subsequent structural modifications. Password management as disclosed in JP-A-2000-181687, in which it suffices to divide an environment to provide environments that users cannot use, cannot resolve this problem. Moreover, in order to avoid the risk of data loss or the like, confirmation of whether the script is composed of data created for setting modification and generating a command group must be guaranteed in the operation.

[0008] In cases where returning to a previous state is desired for some reason, and in situations in which device environments enlarge, for example, storage devices, errors allow past data to be released . With such occurrences, there is also a requirement to return to that environment/structure at a time when the data being released was accumulated.

BRIEF SUMMARY OF THE INVENTION

[0009] This invention overcomes the difficulties of the above prior art, and provides a highly reliable storage control device in which the approval or denial of the execution of commands is determined when a command group relating to configuration information of a storage is received.

[0010] A first embodiment of the invention provides a storage device to receive commands for writing data into and reading data from host devices to control writing and reading of data in storage media. The storage device comprises a service processor for setting storage device configuration information, and a terminal device connected to the

service processor a private line for sending a command group received from an operator, and related to the storage device configuration information, to the service processor. The service processor comprises a means for determining approval or denial of execution of the command group.

[0011] In addition, the terminal device in the invention sends encryption command information, created by encrypting the command group with a secret key, together with the command group. The determining means decrypts the encryption command information to determine whether the command group obtained by the decryption corresponds to the received command group, and executes the command group if correspondence is ascertained.

[0012] A method of setting configuration information for a storage device configured in the above manner includes sending the command group via the terminal device, receiving at a service processor the command group sent, and approving or denying execution of the command group, and executing the command group when approval of execution is determined in the second step.

[0013] More preferably, in a device thus constituted, the invention provides a step of using a secret key to encrypt the command group prior to sending it to the service processor to generate encryption command information. The encryption command information together with the command group is sent to the service processor. Then the processor decrypts the received encryption command information to generate the command group and compares the received command group with the command group obtained by the decryption for correspondence to each other, thus determining approval or denial of execution of the received command group. When the command group obtained by the decryption and the received command group correspond to each other, the processor executes the command group.

[0014] According to the invention, modification of much configuration information can be collectively made with the use of a script sheet. Other devices also read a script sheet whereby such settings can be applied to a plurality of storage devices. Since the storage device does not modify its structure for script sheets other than a previously authenticated one, the structure of the storage device can be modified by information, set by a reliable operator and an operation program, whereby providing a safe setting modification environment free from mistakes is made possible.

[0015] In addition, since communication is possible with an input interface separate from an interface connected to the host computer, settings can be made without connection to

the host computer. Even when the operating system (OS) of the host computer is not started , a script sheet makes it possible to collectively set storage device configuration information and to make a setting for connecting the storage device and the host computer itself.

[0016] Furthermore, unlike a device in which the operational environment is switched over by a multiplicity of operators, the history of the operational environment and the device configuration information is accumulated in a device for continuous information processing, so that a means can be provided that is effective when a state is restored after an error is generated and when the reason for error generation is traced. An advantageous results in that an operator can execute many operations at a time. Other features and objects of the invention than those described above will become apparent when reading the descriptions herein with reference to the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

[0017] Figure 1 is a block diagram showing a configuration of an embodiment of the invention;

[0018] Figure 2 is a flowchart illustrating an exemplary processing flow according to the invention;

[0019] Figure 3 is a flowchart illustrating a procedure for creating a configuration information modification script sheet with a GUI and making its digest, according to the invention;

[0020] Figure 4 is a flowchart illustrating a procedure of creating a script sheet and its digest, according to the invention;

[0021] Figure 5 is a flowchart illustrating a procedure for using and preserving a script sheet according to the invention;

[0022] Figure 6 is an illustration showing an example of a format sheet according to the invention; and

[0023] Figure 7 is a flowchart illustrating a procedure for returning to a previous configuration information setting, according to the invention.

DETAILED DESCRIPTION OF THE INVENTION

[0024] Figure 1 is a block diagram showing connections in a storage system 100 for information processing devices. A storage device 108 is connected to a host computer 101 via a fibre channel 103 to input and output data to storage media 105 on the basis of a

command from host computer 101. An application program 102 is software for operating the host computer.

[0025] A storage management terminal 109A is connected via a local area network (LAN) to create configuration information for storage device 108 to send to the storage device. "Storage configuration information," as used herein, means various settings in a storage device and includes, for example, setting of logic volumes virtually divided on storage media 105.

[0026] In this example, storage management terminal 109A is connected to storage device 108 using a TCP/IP protocol. Control information from storage management terminal 109A is sent and received between a service processor 106 loaded in storage device 108 and a GUI application 110 operating in storage management terminal 109A. In this embodiment, a further storage management terminal 109B is connected to service processor 106 via a storage management interface 111.

[0027] An operator sends control information, such as configuration information or the like, to service processor 106 in storage device 108 via storage management terminal 109B, and service processor 106 updates storage management data 107 on the basis of the control information and resets the structure of the storage device. Meanwhile, the service processor monitors the operating condition of the storage device in order to send the information to storage management terminals A and B via storage management interface 111 and a TCP/IP 104. An operator can recognize from the information received at storage management terminals A and B whether the optimum structure of the storage device has been set.

[0028] Conventionally, the configuration information of storage device 108 has been set on a GUI screen 110 displayed on storage management terminals. According to the invention, however, a script sheet is used in place of, or in addition to a GUI screen. A script sheet, as referred to herein, means a command group with which a service processor sets the structure of a storage device, and which may contain variables required for execution of commands. Such a command group is also called a script definition file, and a processor sequentially reads commands defined in a script to interpret and execute the commands.

[0029] Specifically, script language is used to describe the commands and parameters required for modifying the setting of the storage device structure, which is created by a GUI application program for structure modification operating on storage management terminal A or B. The service processor reads the script sheet to interpret and execute the commands

described therein. As shown in Figure 1, script sheet 110 is a group of operations relating to the addition and deletion of such path definition information as 110-(a) and 110-(b) for connection of host computer 101 and the logical volumes of storage device 108. Also described in the script sheet are operations 110(a) to 110(d) for setting and changing addresses of Fibre Channel Ports or for toggling the Fibre Channel Ports' security switches On/Off.

[0030] Script sheet format 111, shown in Figure 1, is an example in which a setting procedure is written that describes respective settings as argument parameters for setting nomenclatures and information required for the setting thereof. Script sheet 110 in this example has the following meanings. In 110-(a), a path defined in Logical Unit Number (LUN) 1 of Fibre Channel Port 1 is deleted. In 110-(b), the path defined in LUN 1 of Fibre Channel Port 2 is added, and the ID of the logic volume to which the host computer has access is designated 1-1. In 110-(c), the Fibre Address of Fibre Channel Port 1 is designated E3, and in 110-(d), a security switch of Fibre Channel Port 2 is turned Off to impose restrictions on access from the host computer. Such descriptions enable modification of the structure in large quantities to be read and executed by the CPU of a computer for processing, instead of being set through a GUI operation by an operator using key input and a mouse.

[0031] Script sheet format can be dealt with through division into multiple sections and comprehensible descriptions. More specifically, a system referring to sub-script sheets for every setting nomenclature and defining the argument parameters required for operations may serve. Although this system involves numerous sheets, description is facilitated because the parameters are regularly described with every setting.

[0032] Block 601 is an example of a script sheet structure affording convenience when commands described in a script sheet include numerous argument parameters. This example is effective when the same setting is frequently used for argument parameters. Script sheet 603 defines the command "delpath" as deletion of paths communicated to the respective logic units of a port group described in a subsequent text file "eletePathe.txt". In this example, the embodiment is configured such that information in a script sheet is not sent to storage device 108 via fibre channel 103 but is physically interpreted by the service processor through storage management interface 111, which constitutes a logically different interface. That is, because the system of setting via fibre channel 103 cannot use a script sheet unless fibre channel 103 connects to the storage device and host computer 101, setting

configuration information is impossible when host computer 101 is not started up and when an operating system of host computer 101 overruns.

[0033] In addition, troublesome settings, such as setting addresses of ports for connection in large quantities, which should be made by connection to host computer 101, are impossible with a method of interpreting and executing a script sheet on the assumption that operations using host computer 101 are performed. Through storage management interface 111, script sheet 110 enables setting irrespective of connection or nonconnection to the host computer connected to storage device 108.

[0034] Whereas the script sheet 110 enables setting in large quantities, the script sheet itself constitutes text data, which can be easily tampered with, and it is difficult to determine whether tampering has occurred. Moreover, when an error is present in a script file, once set, it is difficult to return the file to an original setting. In a preferred embodiment, the following device is provided to solve the problem. A structure modifying module composed of storage management terminal 109A and service processor 106 is configured to include a module for rejecting script sheet 110 unless electronic authentication has been made; and a module for managing history information when configuration information is modified in the script sheet.

[0035] The module for rejecting an unauthenticated script sheet comprises the steps of creating a script sheet, which is guaranteed by a reliable operator as having a normal structure, and guaranteeing that no tampering has occurred before input into the storage device. Figure 2 shows a specific processing flow. An operator creates a script sheet (201), confirms that the contents of the sheet are valid (202), and starts an electronic authentication processing to guarantee accuracy when the contents are valid.

[0036] In this example, validity of data may be configured such that a script sheet is temporarily set to the service processor and a microprogram loaded on the service processor checks the contents of the script sheet comparing them with a physical structure of the storage device, or validity verification can be configured such that the consistency of data is checked by reliable software, which is software created by a person having a secret key in the subsequent encryption processing.

[0037] Checking of consistency also means checking for the presence or absence of a structural inconvenience that, for example, a newly set logic volume is one, the setting of which is impossible on an actual, loaded storage medium. It is also possible to check whether information set in the script sheet meets a customer's requirements (an electronic file according to specifications required by a customer, or the like). In this case, it suffices to

determine whether a file, which a customer creates by writing data into an essential column, is consistent with the contents of a script sheet created by an operator.

[0038] When the validity of a script sheet is confirmed, authentication processing is executed to show that the script sheet is correct. Creation of a script sheet, confirmation of validity, and authentication processing are carried out using storage management terminals A and B, but may be but may be executed using other information processing devices. Subsequently, a procedure is shown for confirming that a script sheet, which has been created by valid processing, as described above, and the script sheet validated, is reliable.

[0039] In the example, a digest file, in which a script sheet is condensed, is created (203). A digest file depends upon information in a script sheet and is created for that script sheet, for which validity checking was completed in STEP 202, to show that the validity of the script sheet has been guaranteed in STEP 202. The digest file may be created by compressing a script sheet. A digest file is created from a script sheet. Accordingly, the digest file serves at least as a standard for termination of validity checking (203).

[0040] Next, an operator who intends to set the structure of storage device 108 using a digest file, creates the file by encrypting the digest file with a secret key (204). Because a file obtained in STEP 204 cannot be created by anyone other than an operator having a secret key, it can be considered an electronically signed file.

[0041] An operator joins the electronically signed file and a script sheet together to make a file (205) to send to service processor 106 of storage device 108 (206). In this example, an electronically signed file, that is, the file obtained by encrypting a digest file in which a script sheet describing a command group is compressed, and the script sheet are sent as a single file, but the script sheet and the electronically signed file may be separately sent to service processor 106.

[0042] Service processor 106 having received the electronically signed file creates a digest file from the script sheet (207), and then creates another file by decrypting the electronically signed file (encryption) with a public key. Service processor 106 compares the digest file created by the service processor and the file obtained by decrypting the encrypted electronically signed file with a public key (209).

[0043] That is, service processor 106 compares the digest file created from the script sheet and a digest file created by an operator who has a secret key. When the contents of the both files correspond to each other, it can be guaranteed that a sheet, which meets a

customer's requirements and does not cause any inconvenience in setting a new or modified structure, has been sent from a reliable operator.

[0044] Once the script sheet is guaranteed in STEP 209, service processor 106 interprets the script sheet to execute the setting of storage device 108 in accordance with the configuration information of the script sheet (210). Encryption of a script sheet, or a file obtained by creating a digest of a script sheet, is explained below.

[0045] A secret key system uses a technique in which an operator or a control device having a secret key (cord) uses the cord to encrypt information, and the encrypted information cannot be decrypted by operators or control devices other than those having the same secret key. By performing encryption in this system subsequent to the completion of validity checking, execution of commands in the service processor is permissible only with respect to that script sheet, which has been created by an operator having a secret key, thereby eliminating use either of script sheets that have been tampered with, or of inappropriate script sheets in modifying configuration information.

[0046] In addition, although the procedure of creating a digest file for encryption has been described in the example, there is a method of directly encrypting a script sheet with a secret key in which a digest file is not created. In this case, encrypted information and a script sheet are sent to service processor 106, which decrypts the encrypted information with a secret key to create a script sheet, and compares the created script sheet and the received script sheet. In this way, the script sheet can be recognized as valid and having been sent through a reliable procedure.

[0047] When a digest file is created, in addition to reducing the transmission and reception time between storage management terminal 109B and service processor 106, which the command group is being transmitted becomes easy to recognize by creating the digest file in a manner to indicate a feature of the equipment structure.

[0048] By providing two stages, that is, the step of compressing a script sheet as a digest and the step of encryption with a secret key as in the example, indicates that a compressed script sheet is one having undergone validity checking, so that, for example, an operator can perform management steps in which the most appropriate digest file is extracted from a multiplicity of digest files and encrypted with a secret key owned by the operator, to be sent to service processor 106.

[0049] In addition, by specifying an operator or operators having a secret key, only a reliable operator/operators or enterprise/enterprises can provide a script sheet in operation.

Moreover, a reliable operation may be configured to apply automatic encryption with a secret key after validity checking is performed not only by an operator but also by a reliable enterprise, or by reliable software.

[0050] Figure 3 shows an example in which an electronic signature is applied after a software program checks configuration information data modified by the GUI operation performed by an operator and guarantees that the data is consistent. More specifically, an operator carries out an operation using a GUI screen for setting configuration information of storage management terminal 109A. The GUI program has a previously loaded consistency monitoring function to prevent an inappropriate setting in a storage device, and is configured to impose restrictions such that logic volumes of different sizes are not selected as a pair. In this manner, the contents of a script sheet can be guaranteed by applying a secret key only to a script sheet created in a setting operation with specified software and encrypting the script sheet with a secret key.

[0051] Figure 4 shows an example in which information is read from a database in which configuration information is recorded via storage management terminal 109, and a software program checks the consistency of data in the read information to convert the information into a script sheet on which to apply an electronic signature. In this example, when the database itself is reliable, a method of checking configuration information reliability by checking whether information has been taken out of the database may be used.

[0052] A mechanism for managing history information when configuration information is modified in a script sheet is now described in detail with reference to the processing flow shown in Figure 5. With this method, it is possible to return to a desired state when returning to a past state becomes necessary after modification in large quantities has been made by a script sheet.

[0053] First, data described in a script sheet is read by service processor 106 (501). Then, service processor 106 interprets the commands that were read in STEP 501, and executes the processing therefor (502). When processing is successful, the script sheet is preserved in storage management data 107, as a script having proven success, and, at the same time, a history management number and user ID are registered in the sheet so that when a setting was created or modified and the source of that setting can be known.

[0054] Data used in this registration can also be input by an operator when the script sheet is read. Moreover, when the script sheet is created, operating software therefor can be used to preserve the data in the script sheet. When a request (707) for modification of a past

structure is made, as shown in Figure 7, an operator inputs the history management number corresponding to the configuration information so that the service processor can execute the structure modifying processing described in the related script sheet, thus enabling returning to the original configuration information.

[0055] More specifically, an operator inputs a history management number via storage management terminal 109. Service processor 106 refers to storage management data 107 in storage device 108 to read a script sheet corresponding to the history management number to execute processing therefor. Then, data for the date and hour the script sheet was executed, is added to the script sheet to be accumulated as storage management data. In addition, the script sheet referred to here is a group of information and commands for modification of configuration information and various configurations can be designed.

[0056] In other words, according to the invention, when a command group with which configuration information is modified and the parameters required to execute the commands by the service processor from an outside source, the service processor can check the validity of the commands to determine whether to execute them.

[0057] The command group is also encrypted to create encryption command information; both the encryption command information and the command group are sent to a processor for execution of the processing; the processor having received such information decrypts the command information and compares it with the received command group; and on the basis of those results the processor determines whether the received command group can safely modify the configuration information of the device.

[0058] By providing a correlation between commands used in script description and commands, which the service processor actually interprets and executes, an instruction in that expression, which an operator can easily understand, can be given to the service processor. Furthermore, taking account of an unauthorized operation presumed with achieving convenience through such simplification, the service processor is configured to incorporate processing for determining whether the script sheet is safe for the device. When the script sheet has been authenticated, the service processor receives the commands to process the modification of a structure.

[0059] Although the preferred embodiments of the invention have been described in detail, it should be understood that various changes, replacements, and modifications may be made without departing from the spirit and scope of the invention described in the appended claims.